Research in Complexity Theory and Combinatorial Algorithms

Since October 1, 1979, research in Complexity Theory and Combinatorial Algorithms at the Department of Computer Science at the University of Illinois was supported by the Office of Naval Research under contract No. ONR N00014-79-C-0775. During this period of time, research work was carried out in the areas of Computational Complexity Theory, Scheduling Algorithms, Graph Algorithms, Dynamic Programming, and Fault-Tolerance Computing by C. L. Liu, Jane W. S. Liu (principal investigators), and J. R. Egan, M. Ghiassi, J. L. Lewandowski (graduate research assistants). We summarize here our accomplishments and our future plans, and we wish to request continued support for the period of October 1, 1980 - September 30, 1982 from ONR for research in these areas.

I. Scheduling to meet deadlines. The problem of scheduling jobs to meet their deadlines was studied. Given a set of jobs each of which is specified by three parameters, ready time, deadline, and computation time, we want to schedule them on a computer system so that, if possible, all deadlines will be met. Furthermore, if indeed all deadlines can be met, we want to know the possibility of completing the executing of each job so that there will be a "slack time" between the time of completion and the deadline. In particular, the following model is used:

(i) There is a single processor in the computing system.

(ii) Each job consists of an infinite stream of periodic and identical requests. A request is ready when it arrives and should be completed prior to the arrival of the next request of the same job.

(iii) The execution of a job can be interrupted and be resumed later on.

A scheduling algorithm, known as rate monotonic scheduling algorithm is employed [Liu]. In this algorithm, higher priorities are assigned to requests with shorter request periods. In other words, a request of a job with shorter request period will always preempt a request of a job with longer request period. For a set of n jobs, let $T_1, T_2, \ldots, T_n$ denote their request periods, and $C_1, C_2, \ldots, C_n$ denote their computation times. The sum $\sum_{i=1}^{n} C_i/T_i$ is called the utilization factor of the set of jobs and is a measure of the percentage of time (on the average) the jobs will occupy the computer system. We obtained the following results:

Theorem: A set of n jobs with a utilization factor less than or equal to $n(2^{1/n} - 1)$ is always schedulable by the rate monotonic scheduling algorithm.

Theorem: A set of n jobs with a utilization factor less than $n(2^{1/n} - 1)$ is always schedulable by the rate monotonic scheduling algorithm so that every request of each job has a non-zero slack time. Furthermore, for each job, the slack time of any request is larger than or equal to 0.207q where q is the length of the last quantum of time allocated to the first request of that job.

These and other results are included in a paper by C. L. Liu, J. W. S. Liu, and A. I. Liestman (see Appendix) which has been submitted for publication.

II. Models for Performance Evaluation of Computing Systems. To evaluate the merits of using special purpose processors in multiprocessor computing systems, we analyzed several models, including both deterministic and probabilistic ones, and obtained different performance measures which can be used as criteria of comparison among different configurations of multiprocessor systems. Using a deterministic model, we show that when suboptimal demand scheduling strategies are employed, the performance of a system deteriorates very rapidly when there is a significant disparity in speeds among the processors in the system. Intuitively, we can envision the loss in scheduling flexibility when a system becomes highly heterogeneous. To determine the minimum ratio between the speeds of special purpose processors and general purpose processors so as to achieve a fixed over all system capabilities, several approximate queueing models were studied.

A paper containing these results entitled "Performance Evaluation of Multiprocessor Systems Containing Special Purpose Processors" by J. W. S. Liu, C. L. Liu, N. Miyazaki, H. Yamazaki will appear in a book on Systems Modelling and Performance Evaluation edited by E. Gelenbe (see Appendix).

III. Generation of k-ary trees. The problem of designing algorithms for generating ordered trees was studied. In some application problems, one might need to examine all k-ary trees with a certain number of internal nodes (for example, exhaustive search of some structural properties of trees, and simulation involving trees). There are also situations in which one might need to generate a "random" k-ary tree. As in all cases concerning the generation of combinatorial objects, the three problems usually encountered are (i) design an algorithm for generating all k-ary trees, (ii) design an algorithm for determining the rank of a given tree, (iii) design an algorithm for generating a tree when its rank is given.

In order to solve these problems, we need to adopt first a convenient way to represent k-ary trees. We discovered a new representation of k-ary trees which is quite useful for our purpose. We traverse a k-ary tree in the root - first subtree - second subtree - ... - k^{th} subtree order and label the internal nodes with the integers 1, 2, ..., n in the order they are visited. We then traverse the tree again in the order first subtree - root - second subtree - root - ... $(k-1)^{st}$ subtree - root - k^{th} subtree and read off the label of an internal node every time it is visited. We thus obtain a permutation of a multiset consisting of (k-1) 1's, (k-1) 2's, ..., (k-1) n's. As it turns out, such a permutation provides a unique representation of a k-ary tree. Using such a representation, we designed (i) a generating algorithm, (ii) a ranking algorithm, and (iii) an unranking algorithm for k-ary trees.

Furthermore, our ranking and unranking algorithms can be applied to the cases where k-ary trees are represented in different ways, namely, sequence of 0's and 1's, and sequence of level numbers [Rus], [Zak].

A paper containing some of these results by C. L. Liu was presented at the Fifth Annual Symposium on Trees in Algebra and Programming held on February 21-23, 1980 in Lille, France. Another paper thatincludes all the details has been prepared for submission for journal publication. (See Appendix.)

IV. <u>A boundary between P and NP-Complete Version of a Problem</u>. Since the introduction of the notion of NP-completeness in the study of Complexity Theory about ten years ago, there has been an explosion of research activities in the area. In particular, the collective effort of many researchers has led to the discovery of a very large number of problems that are certified to be NP-complete. However, there is an interesting question about which very little is known, namely, what makes a problem NP-complete? In order to gain further insight into the characterization of NP-complete problems, we studied several problem areas in which there is a spectrum of problems whose complexities vary with some "small" variations on the assumptions. We studied these spectra of problems, and we hope that further understanding on the "boundary" that separates those problems in P and those problems that are NP-complete will lead to further understanding on the characterization of NP-complete problems.

A problem, which is referred to as the k-color Path Problem, can be formulated as follows: Let G be a given graph with its vertices colored with k colors. We want to determine whether there exists a path from a source to a sink that contains at least a vertex of each color. When k is not fixed, this problem is clearly NP-complete since for $k = |V|$, the k-color Path Problem becomes the Hamiltonian Path Problem. On the other hand, when $k = 1$, the problem is obvious polynomial. Somewhere in between, there exists a boundary separating the two classes of problems. As to the case where G is a directed graph, it can be shown that the problem is NP-complete even when $k = 2$. As a matter of fact, the problem remains to be NP-complete when the additional condition that each color is used to color at least n of the vertices for a fixed n is imposed. On the other hand, we have yet determined the complexity of the problem when the condition that each color is used to color at least $c|V|$ of the vertices for a fixed fraction c is imposed.

Another problem that we studied is the Subgraph Homeomorphism Problem (SHP). Given two graphs G and H, the problem is to determine whether there exists a mapping f from the set of vertices of H onto the set of vertices of G such that for every edge (v_i, v_j) in H there is a corresponding path between $f(v_i)$ and $f(v_j)$ in G. Several variations to the problem are possible. For example, the graphs G and H can both be directed or undirected, the paths in G can be either vertex-disjoint or edge-disjoint, the pattern graph H can be either fixed or varied for different G.

When G and H are directed graphs, Fortune, Hopcroft, and Willey [For] have shown that the SHP is NP-complete even when H contains only two vertex-disjoint edges. On the other hand, it is easy to see that SHP is in P when H does not contain two vertex-disjoint edges. Thus, the SHP is completely settled for the case where G and H are directed graphs.

When G and H are undirected graphs, the SHP is NP-complete when H is not fixed because it includes either the Hamiltonian Circuit Problem (for the vertex-disjoint version of SHP) or the Multi-Commodity Flow Problem (for the edge-disjoint version of SHP) as a subproblem. However, when H is a fixed pattern graph (i.e., H is no longer part of the input to an algorithm that solved the SHP), the complexity of the problem is not yet completely settled. For example, polynomial algorithms for solving the SHP when H is a triangle or when H consists of two vertex-disjoint edges have been found [LaP][Shi]. Thus, the SHP provides another good problem area for studying the boundary between problems that are in P and problems that are NP-complete.

We studied and simplified an algorithm due to LaPaugh and Rivest [LaP] which solves the SHP when H is a triangle. We also obtained an algorithm for the case when H is quadrilateral. We plan to continue to pursue our line of approach for the case that H is a k-sided polygon.

For the details of our work, see the enclosed interim report by J. R. Egan in the Appendix.

V. Orientation and Partition of the Edges of a Graph. Let G = (V, E) be an undirected graph. A very general question one can ask in whether and how the edges in E can be oriented so that the resultant directed graph, denoted $\vec{G}$, has certain property P. A problem of this type was studied by H. Robbins [Rob] which asked whether the edges in E can be oriented such that $\vec{G}$ is strongly connected. A possible physical interpretation of the problem is to consider the edges in E as two-way streets in a city and to ask whether it is possible to change all two-way streets to one-way streets such that it is still possible to go from any point of the city to any other point. Another interpretation is to consider the edges in E as two-way airlines service among cities and to ask whether it is possible to reduce the service to one-way service so that every city is still reachable from any other city. In [Rob], it was shown that a necessary and sufficient condition for the existence of such an orientation is that G is connected and is bridge-free (i.e., 2-edge connected.) This result was later generalized by C. Nash-Williams who showed that if G is 2k-edge connected, then there is an orientation of the edges of G such that $\vec{G}$ is k-arc connected.

One direction of generalization carried out by J. L. Lewandowski [Lew] is exemplified by the question: Given $G = (V, E)$ and two subsets A and B of V, is there an orientation of the edges in E such that for each vertex v in V, there is a directed path from every vertex in A to v, and a directed path from v to every vertex in B? This problem can be interpreted as having certain cities (those in the subset A) which must be able to reach every other city in V and certain other cities (those in the subset B) which must be reachable from every other city in V. As it turned out, this problem is easily reduced to Robbins' problem and is therefore in P. A slight variation of the problem leads to some interesting questions: Let A be a subset of V, we want to know if there is an orientation of the edges such that there is a path from every vertex in A to every vertex in $\bar{A}$ (i.e., $V - A$). (Clearly, we could also ask for corresponding paths in the opposite direction; namely, from every vertex in $\bar{A}$ to A.) It is easily seen that in the case where A contains only a single vertex, it is necessary and sufficient for G to be connected. However, we are still looking for a general characterization of the pair (G, A) when such an orientation is possible.

Chvátal and Thomassen [Chv] studied the problem of orienting the edges of a graph $G = (V, E)$ so that the radius (or diameter) of $\vec{G}$ can be as small as possible. They showed that if the radus of G is r, then it is always possible to obtain $\vec{G}$ the radius of which is atmost $\frac{1}{2}(r^2 + r)$. Furthermore, this bound is best possible. B. Eberle [Ebe] discovered a polynomial time algorithm, $O(|V|^4)$, that yields $\vec{G}$ with its radius upperbounded by $\frac{1}{2}(r^2 + r)$.

Frank and Gyárfás [Fra] studied the orientation problem with the requirement that $\vec{G}$ possesses one or more of the following properties:

(i) $\vec{G}$ is strongly connected.

(ii) $\vec{G}$ has a directed tree with a given root.

(iii) The outdegrees of $\vec{G}$ lie between given upper and lower bounds.

Such a problem is motivated by a study of the flowcharts of computer programs. A large class of flowcharts corresponds to a class of directed graphs in which (i) there is a vertex that can reach every other vertex through a directed path (this vertex corresponds to the starting point of a program), (ii) the outdegree of each vertex is atmost 2 (this condition amounts to limiting all branching operations in a program to two-way branching).

Another problem concerning the orientation of the edges a graph is related to our study of the diagnosibility of digital system (see VII). Given an undirected graph G = (V, E), we want to know how the edges in E can be oriented so that the <u>one-step diagnosibility</u>, t_0, or the <u>sequential diagnosibility</u>, t_r, can be as large as possible.

A variation to the graph orientation problem is that of coloring the edges (which is equivalent to partitioning the edges into classes) so that the resultant graph will have some desired properties. For example, one might ask whether it is possible to color the edges of a graph with two colors so that there will be two monochromatic spanning trees. Clearly, this is equivalent to asking whether a graph can be decomposed into two edge-disjoint connected subgraphs. W. Tutte has obtained an elegant combinatorial characterization of the decomposibility of a graph into k connected subgraphs. We intend to carry out a thorough study of this problem from an algorithmic point of view. A slight generalization of the problem leads us immediately to the domain of NP-completeness: Let A and B be two disjoint subsets of V. Let v be a vertex in V. The problem of determining whether the edges in E can be colored with two colors so that there is a red path from v to every vertex in A and a blue path from v to every vertex in B is NP-complete. (For a proof of the result, see enclosed interim report in the Appendix.) A natural interpretation of this problem is to consider v as a source that supplies two kinds of commodity and to consider the edges as pipelines,each of which can carry either one of the two kinds of commodity. Our question is whether we can designate the pipelines in such a way that all the vertices in A will receive one kind of commodity and all the vertices in B will receive another kind of commodity.

Another partitioning problem in which there have been good results, but with some remaining open questions is the problem of disjoint connecting paths. That is, given a graph G, and a set of disjoint vertex pairs $\{(s_1, t_1), (s_2, t_2) \ldots (s_k, t_k)\}$ is there a k-coloring of G such that there is a path of color i between s_i and t_i for all $1 \leq i \leq k$. For k = 1 the result is trivial, and for k = 2, the problem was shown to be solvable in polynomial time by Shiloach. However, the general problem is NP-complete but the complexity for any fixed $k \geq 3$ is unknown.

We plan to continue our work in the various aspects of this problem area in the forthcoming year.

VI. <u>Dynamic Programming</u>. In a multi-stage dynamic programming problem, an optimal policy consists of a sequence optimal decisions at all the stages. We are interested in studying the effect of a suboptimal decision at one or more of the stages. In particular, for a given multi-stage decision problem if it is known that a certain suboptimal decision will be made at one of the stages, we want to know the over all effect if it takes place at an earlier stage or at a later stage. This depends, of course, on the decision problem itself as well as the nature of the sub-optimal decision. (For example, in a multistage resources allocation problem, a suboptimal decision can be one that allocates a fixed amount of the resources or one that allocated a fixed proportion of the resources available, and so on.) At the moment, we are working on some general characterization of dynamic programming problems as exemplified by the following results:

Consider the problem of minimizing the quantity

$$g(x_1) + g(x_2) + \ldots + g(x_n)$$

subject to the constraint

$$x_1 + x_2 + \ldots + x_n = a$$

The problem can be formulated as a multi-stage dynamic programming problem in which

$$\begin{aligned} f_n(a) &= \min_{x_1+x_2+\ldots+x_n = a} [g(x_1) + g(x_2) + \ldots + g(x_n)] \\ &= \min_y [g(y) + f_{n-1}(a-y)] \end{aligned}$$

Suppose instead of making optimal decisions during the n decision steps, suboptimal decision was made in one of the steps. The question we have is the effect of such a suboptimal decision on the total return. We obtained:

Theorem: Suppose that during the i^{th} decision stage, instead of choosing an optimal value x_i, a fixed value α is chosen, then the deviation from the optimal value is the smallest possible if the sub-optimal decision was made at the first stage.

Theorem: Suppose that during the i^{th} decision stage, instead of choosing an optimal value x_i, the value $\alpha(a - \sum_{j=1}^{i-1} x_j)$ for a fixed α is chosen, then the deviation from the optimal value is the smallest possible if the suboptimal decision was made at the first stage provided (i) $x_1 > \alpha a$ (ii) the function $g(y)$ is of the form $g(y) = c_k y^k + c_{k-1} y^{k-1} + \ldots + c_1 y$ where all c's are non-negative.

Further investigation of this problem will be conducted.

VII. Fault-Tolerant Computing. We studied a problem concerning diagnosis of digital systems. We use a model that was first introduced by Preparata, Metze, and Chien [Pre]. In this model, a digital system is partitioned into a certain number of units, each of which can be at one of two possible states, fault-free ($\bar{F}$) and faulty (F). A configuration of a system is an assignment of either the fault-free or the faulty state to each unit in the system. We assume that each unit in the system possesses a certain amount of computational resources to enable it to test one or more of the other units in the system. The outcome of a test is a binary signal which depends on the state of the testing and the tested units. In particular, we assume that:

(i) if a fault-free unit is tested by a fault-free unit, a signal 0 will be generated,

(ii) if a faulty unit is tested by a fault-free unit, a signal 0 will be generated,

(iii) if a fault-free or a faulty unit is tested by a faulty unit, either a signal 0 or a signal 1 will be generated. (In other words, the signal generated by faulty testing unit is completely unreliable.)

A diagnosis experiment is one in which every unit tests all the units it is capable of testing once. The outcomes of the tests are referred to as a syndrome. The goal of a diagnosis experiment is to identify one or more of the faulty units in the system. A one-step diagnosis is one in which all faulty units in the system are identified. A sequential diagnosis is one in which at least one faulty unit, if there is any, is identified. For any system, both one-step diagnosis and sequential diagnosis are possible, provided that the number of faulty units does not exceed certain critical value. The one-step diagnosibility of a digital system, t_0, is defined to be the maximum number of faulty units in the system such that for any snydrome corresponding to a configuration with no more than t_0 faulty units, one-step diagnosis is possible. The sequential diagnosibility, t_r, is defined to be the maximum number of faulty units in the system such that for any syndrome corresponding to a configuration with no more than t_r faulty units, sequential diagnosis is possible. The problem of determining t_0 and t_r is, in general, a difficult one [Fuj]. We developed a new technique for obtaining lower bounds on the value of t_r for a class of digital systems. An example of our results can be stated as:

Theorem: Let G be a directed graph that described the interconnection of a digital system. If G contains a 2-star of size k, then

$$t_r \geq \left\lceil \frac{k-1}{2} \right\rceil$$

For further details, see Appendix for the paper by P. Maestrini and C. L. Liu.

References

[Chv] Chvátal, V., and C. Thomassen, "Distances in orientations of graphs," Stanford University Tech. Rep. No. STAN-CS-75-517, August, 1975.

[Ebe] Eberle, B.D., "Algorithms for orienting graphs," M.S. Thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 1979.

[For] Fortune, S., J. Hopcroft and J. Wyllie, "The directed subgraph homeomorphism problem," Theoretical Computer Science, 1980, 111-121.

[Fra] Frank, A., and A. Gyárfás, "How to orient the edges of a graph?", Colloquia Mathematica Societatis János Bolyai, 18 Combinatorics, Keszthely, Hungary, 1976, 353-363.

[Fuj] Fujiwara, H., and K. Kinoshita, "On computational complexity of system diagnosis," IEEE Trans. on Comput., Vol. C-27, 1978, 379-384.

[LaP] LaPaugh, A.S., and R. L. Rivest, "The subgraph homeomorphism problem," Proc. Tenth Annual ACM Symposium on Theory of Computing, San Diego, CA, 1978, 40-50.

[Lew] Lewandowski, J. L., "On orienting graphs," M.S. Thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 1979.

[Liu] Liu, C. L., and J. W. Layland, "Scheduling algorithms for multiprogramming in hard-real-time environment," JACM, 1973, 46-61.

[Pre] Preparata, F.P., G. Metze, and R. T. Chien, "On the connection assignment problem of diagnosable systems," IEEE Trans. Electron. Comput., Vol. EC-16, 1967, 848-854.

[Rob] Robbins, H. E., "A theorem on graphs with an application to a problem of traffic control," Amer. Math. Monthly, 1939.

[Rus] Ruskey, F., "Generating t-ary trees lexicographically," SIAM J. Comput., 1978, 424-439.

[Shi] Shiloach, Y., "The two paths problem is polynomial," Stanford University Tech. Rep. CS-78-654, 1978.

[Zak] Zaks, S., "Lexicographic generation of ordered trees," Th. Comput. Sci., 1980, 63-82.